Management of Dynamic Networks and Services

Olivier Festor, Radu State

Management of Dynamic Networks & Services

A sample of candidate technologies and a case study

Olivier Festor, ***Ph.D., Hab.***

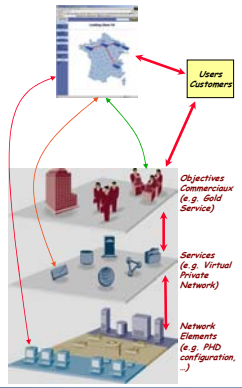
Radu State ***Ph.D.***

MADYNES

The MADYNES Research Team
LORIA – INRIA Lorraine
615, rue du Jardin Botanique
54602 Villers-lès-Nancy
France
{Olivier.Festor,Radu.State}loria.fr

Preamble : Management

- All one does to keep the subject :
  - Healthy (up and running efficiently)
  - Evolving positively
- Fundamental operations
  - ***Monitoring***
    - Observe the subject at work ☺
  - ***Control***
    - Act
    - change or constrain the behavior of the managed subject
  - *Plan its evolution*
- Why ?
  - Service quality
  - Reduce costs
  - With limited risks
    - Keeping the right level of quality



Preamble : "management" wording in the presentation

Management

⇔

Device Management	Network Management
Service Management	Application Management

Configuration	Management
Fault	Management
Accounting	Management
Performance	Management
Security	Management

– Unless explicitly stated otherwise.

Tutorial Outline

- Growing Dynamics and their Impact on Management *(20 minutes)*
  - Dynamics
  - Collapsing time scales
  - The need for automation and autonomy
  - Management solutions for the dynamic world
  - Standardized approaches
  - Emerging alternatives
- JMX : Dynamic Management for/through the Java World *(60 min)*
  - Basic Concepts
  - Dynamic components
    - Attribute, method & notification level
    - Managed Object level
  - Remoting
  - Implementations
  - Application domains & instrumentation patterns
- SyncML-DM : an Approach for Managing Dynamic Devices *(60 min)*
  - Representation Protocol for Device Management
  - Device Management Protocol
  - Standardized Objects
  - Device Management Tree
  - Security for Device Management
- Case studies *(40 min)*
  - Ad hoc networks management
- Conclusion & discussion *(15 min)*

Dynamic systems we are interested in :

- In general :

  « A system that changes its state over time »

  [Jakobson 2001]
- From a management plane point of view :

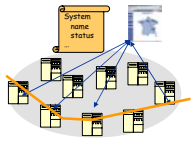
« A system that changes its state over time in such a way that changes cannot be detected or processed efficiently by the management plane »

  - Because they occur too fast
  - Because they occur too often
  - Because they occur too slowly
  - Because their processing requires changes in the management plane

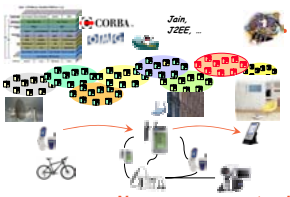
Dynamics Everywhere in Networks and Services !

- Topology related dynamics
  - Ad hoc networks : connectivity, reachability, autonomy
  - P2P networks & overlays : availability, presence, contribution, topology
  - Mobile networks : mobility, power, usage
- Capacity related dynamics
  - Constrained power environments
  - Vertical handoff & Associated QoS variation
- Services & Service Infrastructure level dynamics
  - Software radio, extensible terminals
  - P2P networks & overlays
  - Active networks : protocols change, functionality extension, …
  - Service platforms : service life cycle, hot deployment, update, …
  - Service usage : discovery, trading, late binding, lease, subscribtion
- Device, Network & Service Management Dynamics
  - Online SLA parameters (re)-negociation
  - Management interfaces tailoring
  - …

Collapsing Time scales



- Static Management
  - Manager/MIB/Agent
  - Manual configuration of participating entities
  - Homogeneous management domains
  - Pre-sharing of management information models
  - Size : 1 to a few hundred
  - ***Time Scale***
    - ***Few minutes up to a few month***
    - ***1-8 years for information models (standards)***



Dynamic Management
- Infrastructures, services, usage & mobility
- Multiple, shared evolving administrative domains
- Variable information models and management interfaces
- Negociable & evolutive management functions
- Size : 1 to several millions/billions
- ***Time Scale***
  - ***ms up to a few seconds***

New management solutions are needed!

Management solutions for the dynamic world

1. Change to a static world and keep the management frameworks as they are !
   - 1-6 years to build a management interface
   - **Accept that you will at some time end in state #2 !**

2. Consider management as a useless interface and service and stop working on it !
   - Start thinking about other ways to QoS*ing*
     - Trust webs, in-band probing, signalling, …

3. Make management as dynamic as the entities it manages !
   - Rethinking everything

Mgmt & dynamics (1): the static option

- « Many aspects of dynamic systems are often perceived as static » *[Jakobson 2001]*
  - So lets consider them as static
  - As long as theses dynamic aspects do not alter the external enveloppe.
  - Keep the management frameworks as they are
- Mapping in the networking world
  - Focus on the service or application level only
    - End2End monitoring & performance measurment
  - As soon as something becomes too dynamic : ignore It
- History tells us that this is not a good approach
  - Often micro-dynamics highly impact the world
    - « Butterfly Effect » by E. Lorentz MIT (1961)
    - Maybe not the best example …
  - In highly dynamic environments, even management of the static parts becomes almost impossible using the usual paradigms

Mgmt & dynamics (2): the empty chair option

- Networks and services do not need management anymore !
  - They adapt themselves to the provided conditions
  - 90's ALF philosophy with an extension to the active networks community
- **But «** Nature abhors a vacuum **»**
  - *Benedict Spinoza (1632-1677)*

- **Thus** traditional management will be replaced by **smart solutions**
  - From management to signalling
    - The case of an evolution : ATM VPC, SVC
  - DHCP was an evolution in IP address management
    - but not a sufficient one since DHCP needs to be managed and DNS is required too for the translation (Dynamic DNS is a partial solution)
  - Zeroconf, zerouter are other « management free » efforts within IETF
    - No management entity needed
    - Initially focused on IP address (unicast, multicast) and name assignment
- **Because** Unmanaged entities are part of the **Chaos !**

Mgmt & dynamics (3): make management dynamic

- ***Divide ut imperes!*** *(Divide & conquer !)*
  - *Julius Cesar (102-40 BJC)*
  - Manage smaller worlds to manage each better
    - Get compliant with each degree of encountered dynamicity
  - Build an adequate hierarchy to manage the interactions among the worlds
    - This is NOT an easy task!
- **Foster Management Autonomy & Automation**
  - Avoid human interference
    - **Self-*Anything*** for the management plane
    - E.g. self-configuration of the management plane
  - Avoid time costly operations
    - Like long interface standardization efforts
    - New dedicated management languages & models learning,
    - ….
- **Equivalent to the consequence of the previous option**
- **Accept & Integrate new constraints in the management plane**
  - Low connectivity
  - Shared / cooperative management, new security models, negociation services
  - Unstable domains & entities
  - Evolving and changing management interface

Standardized approaches support

- Mostly focused on Delegation & Distribution
  - RMON & RMON2
  - SNMP's INFORM Service
  - SNMP DISMAN initiative
    - Script MIB, Event MIB, Schedule MIB, Remote Operations MIB,
- Adaptation to emerging constraints
  - Dynamic MIB extensibility support
    - **AgentX** & previous efforts like SNMP-DPI or SMUX
    - **CIM Provider** in the WBEM Architecture
  - Asynchronous messaging
    - **MOM's** gain acceptance as a management communication channel
- Current limits (self-*ing*)
  - Self-configuration
    - Service, network, device & MANAGEMENT PLANE !
  - Self-provisioning
  - Self-instrumentation of services,
  - Self monitoring
  - Self-healing
  - Self-optimizing

Emerging alternatives

- Delegation support and management interactions remain a strong focus
  - Active networks based management is one approach
  - Mobile agents are another solution
  - Dynamic Managed objects loading & distribution become common
- Dynamic instrumentation is growing fast
  - CORBA interceptors
  - Web Services Proxies
  - J2EE management introspection
  - OSG*i* service interface monitoring
- Automation is increasing
  - Closed-loop management processes
    - E.g. Self optimizing performance (smart scheduling in Web Farms, storage networks, …)
  - SLA/SLM automation
    - Formal SLA parameters & automated mapping to monitoring & reporting procedures

Summary

- Automation is the focus of many efforts
  - Autonomic Computing by IBM
  - N1 for Farmed resources by Sun
  - CISCO CNS Configuration Engine & Intelligence Engine …
  - SelfCon initiative (Nortel & Waterloo U.)
  - Self-aware networks (Alcatel)
  - …
- Promising future
  - Several technologies cover parts of the needs
    - But do not cover all needs at the same degree
    - We will examine 2 different ones with different features in the next sections
  - Research proposals in automated management are very promising
- New « to-be managed environments » appear :
  - On demand, self-service, home service platforms and environments
  - Ad hoc, sensor, home, wearable networks
  - Context sensitive services, including management onces

Tutorial Outline

- Growing Dynamics and their Impact on Management *(20 minutes)*
  - Dynamics
  - Collapsing time scales
  - The need for automation and autonomy
  - Management solutions for the dynamic world
  - Standardized approaches
  - Emerging alternatives
- JMX : Dynamic Management for/through the Java World *(60 min)*
  - Basic Concepts
  - Dynamic components
    - Attribute, method & notification level
    - Managed Object level
  - Remoting
  - Implementations
  - Application domains & instrumentation patterns
- SyncML-DM : an Approach for Managing Dynamic Devices *(60 min)*
  - Representation Protocol for Device Management
  - Device Management Protocol
  - Standardized Objects
  - Device Management Tree
  - Security for Device Management
- Case studies *(40 min)*
  - Ad hoc networks management
- Conclusion & discussion *(15 min)*

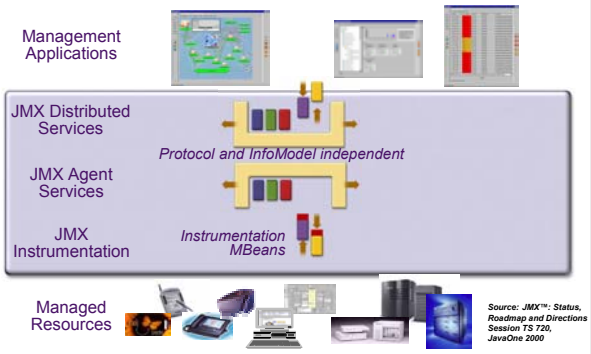
JMX : Java Management Extensions

- Why JMX in a tutorial on Management of Dynamic Networks and Services ?
  1. Used in dynamic service environments
     - Inspired misuses !
  2. Interesting features :
     - Multi-level dynamics
     - Great learning curve
  3. Part of the evolution of the management species
     - Strong heritage from the management roots
  4. Excellent Open Source implementations

JMX : Java Management Extensions

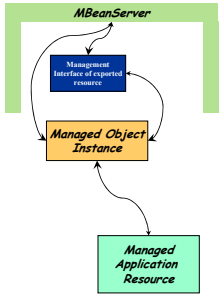
- Java-based Management Agent Architecture
  - Enables Java applications/components to be easily instrumented in Java
  - Fosters the use of the Java technology to manage non-Java environments
- Information Models
  - Set of MBean objects
- Managed Objects
  - Name + MBean
- X.500 like naming
  - *But not exactly the same !*
- Decoupling of instrumentation / export / access protocols

JMX global architecture



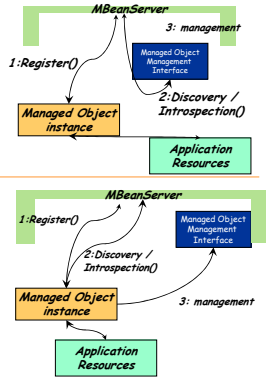
JMX : Managed Objects

- Simplified vision
  A management interface = MBeanInfo
  - Exposed Attributes
  - « invokable » methods on the management interface
- 3 levels of MO programmability
  - Who builds the management interface ?
    - The agent through introspection
    - The application as a demand made by the agent
    - The agent through an external order
  - What is the dynamics of the interface ?
    - Defined at compile-time
    - At run-time
  - Who build the Managed Object that supports the interface
    - Application developper
    - The agent, through delegation



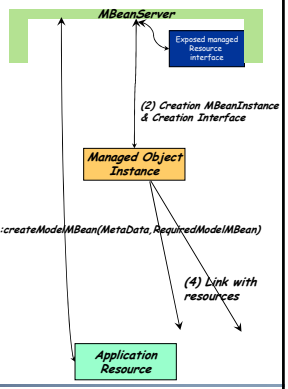
JMX : Managed Objects

- Standard MBean
  - Management interface defined at compile-time
  - Interface discovered by the agent at object registration time
  - + /-
    - Simple model
    - Interface is Static
- **Dynamic** MBean
  - Exposed Management interface is **built at run-time**
  - Management interface delegated by the agent to the MO
  - +/-
    - Coding Overhead
      - MBeanInfo, get, set, invoke, …
    - Context sensitive management interface setup made possible (remember the conditional packages in the OSI management framework ☺)



JMX : Managed Objects

- *Open MBean*
  - *Dynamic MBean with :*
    - *Restrictions on possible types*
      - *Method parameters*
      - *Return types*
    - *Extended MBeanInfo*
- **Model** MBean
  - Resource delegates the « coding » of the MBean to the MBeanServer
  - Interface defined at MO run-time
  - + /-
    - Ideal model for NON-management specialists
    - Marvelous free services :
      - Cache management
      - Persistance
      - Logging, …



JMX : a small Standard MBean example

```
// Interface
public interface PrinterMBean {
public Integer getMaxCopies();
public void setMaxCopies(Integer n);
public PrinterStatus getStatus();
public void reset(); }

// The MBean
public class StandardPrinter implements PrinterMBean
{
  private Integer supCopies = 2;
  private PrinterStatus fStatus;
  ...
  public Integer getMaxCopies()
  {return supCopies;}

  public void setMaxCopies(Integer pSubCopies)
  {fSupCopies = pSubCopies;}

  public PrinterStatus getStatus()
  {return fStatus;}

  public reset() {setStatus("down","other"); sendReboot();}

public Printer()
{ fStatus = new PrinterStatus();
};

 // not visible at the Mgmt Interface
 public Integer getTemerature() {return fTemperature;};
}
```

```
public class MonApplication {

// The agent
private MBeanServer myMBeanServer  =
            MBeanServerFactory.createMBeanServer();

public MonApplication
{
 CommunicatorServer htmlAdaptor = new
                              HtmlAdaptorServer();
try {
      ObjectInstance htmlAdaptorInstance =
       myMBeanServer.registerMBean(htmlAdaptor, null);

      ObjectName mbeanObjectName = new
                  ObjectName("madynes:id=MyFirstMBean");

    myMBeanServer.createMBean(
                "PrinterMBean",
                PrinterMBean);

    }
catch(Exception e) {  e.printStackTrace();}
}
```


JMX : agent services

- MBeanServer
  - MBeans container
  - Ensures MBeans naming & registering
- Advanced services
  - Monitoring
    - Objects attributes
      - Numerical values & character chains
  - Timer
    - Periodical, « one shot »
  - MLet
    - Dynamic loading of MBeans
  - Query
    - Selection operation on MBeans
    - OSI style Scoping & Filtering
  - Relationship
    - Setup and maintain relationships among managed objects

JMX : event model

- Events are part of the JMX model
  - Are thus well integrated in the framework
  - Are used in many fundamental services of the framework
- Java's Event/Listener Model
  - Emitters
    - Broadcaster interface
  - Event consumers
    - Subscription & listeners
- Done within one MBean Server

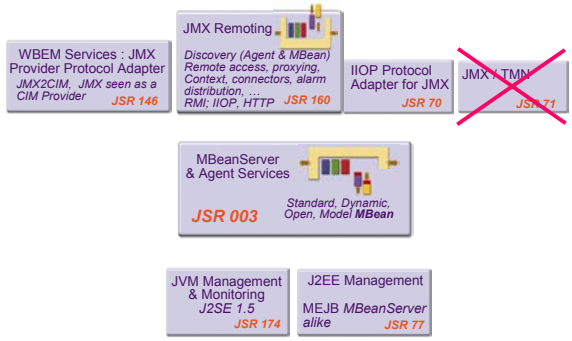
```
public class StandardPrinter implements PrinterMBean,
      NotificationListener
{
 public void handleNotification(Notification notification, Object
      handBack)
 {
  System.out.println(notification.getMessage();
 }
 .....
 public PrinterMBean(SystemMbean systemBean)
      {....
       NotificationFilterSupport nf = new
                                  NotificationFilterSupport();
      nf.enableType(new String(« PC.alarm »));
      systemBean.addNotificationListener(this,nf,hb);
      }
}
```

```
public class SystemMBean extends NotificationBroadcasterSupport
      implements MyPCMBean
{
}
.....
```

JMX : event components

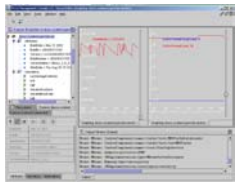
- Notification (CL can be subclassed)
  - type: character string which defines the notification type ex. fr.loria.office.fire
  - seq number: in the context of the source
  - Time stamp
  - message: string describing the cause
  - userData: additional data provided by the sender (Java Object)
- NotificationListener (IF)
  - handelNotification(Notification n, Object handback)
- NotificationFilter (IF)
  - isNotificationEnabled (invoqué par le broadcaster)
- NotificationBroadcaster (IF)
  - getNotificationInfo: lists all notifications that the source can send
  - add/RemoveNotificationListener
  - sendNotification

JMX & related proposals in the JCP



JMX : platforms

- Open Source platforms
  - MX4J (mx4j.sourceforge.net)
    - Full toolkit integration
    - Several extensions
      - Additional connectors
      - Extended Meta-models (MBeansInfo) in standard MBeans
    - Apache License
  - Open JMX : MX4J predecessor
  - JBOSSMX :
    - Core of the J2EE JBOSS platform
    - A genious « misuse » of JMX
  - *XMOJO (xmojo.org, LGPL)*
- Others
  - TMX4J
    - Tivoli, alphaworks
  - JMXRI
    - Reference implementation
  - JDMK
    - Sun's offer
  - Advent Net JMX Studio
    - Development environment & applications
  - MC4J (mc4j.sourceforge.net)
    - JMX management console
    - Compatible with :
      - MX4J
      - JBOSSMX
      - JDMK



JMX Users *(source JMXperience)*

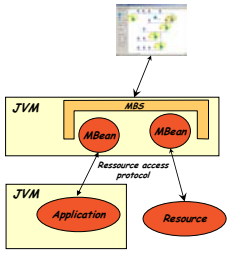
- AdventNet: Agent Toolkit - Java/JMX Edition + Manage Engine + Middleware Manager + Middleware Manager WebLogic Edition + WebNMS
- BEA: WebLogic
- Compiere Open Source ERP & CRM
- Critical Path: Internet File Server, Presentation Server, Registered Mail Server, System Console
- CSC (Scandinavia): LABKA II
- Dirig Software: Dirig Agent
- Hewlett Packard: Core Services Framework + HPAS + OpenView
- IBM/Tivoli
- IBM: Web Services Toolkit 3.1 , + WebSphere Business Components + WebSphere Business Integrator +WebSphere Voice Server
- Innovative Systems Design: ITVerify
- IONA Technologies PLC: iPortal + Orbix + Orbix E2A XMLBus Edition Technology 2.0
- iReasoning Networks: iReasoning JMX SNMP Agent BUILDER
- The Jakarta Project (Apache): Phoenix
- JBoss
- Log4j
- Lutris: EAS
- Macromedia: FlashMX, JRun 4
- Manage.com: FrontLine Java Management Edition (JME)
- Media style GmbH
- Misys International Banking Systems Ltd: Meridian
- ObjectWeb: JOnAS 2.5, JORAM 3.1
- Pramati Technologies
- Resonate Inc: Resonate Commander
- Schmid Telecommunication: Pegasus Element Manager
- Sonic Software: SonicXQ
- SpiritSoft: SpiritWave 5.1 + SpiritWave Integration Server
- Sun Microsystems: Java Dynamic Management Kit (JDMK) + Netra CT Managed Object Hierarchy (MOH) + Netra HA Suite + Netra T1 + DReAM, Distributed Resource Allocation Manager + SunONE Application Server + SunONE Portal Server
- Sybase: EAServer 4.0
- TCC: Rexip AppServer 1.0
- Tomcat
- Wily Technologies: Introscope
- XadrA's VelocityAdaptorServer
- Zareus, Inc: Zareus Application Platform

JMX Deployment Models *[KregerHW 03]*

- MBean
- JVM
- MBeanServer
- Application

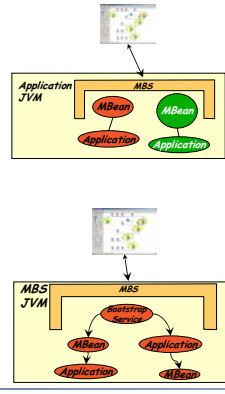
Relationship ?
Who owns what ?
Who initiates what ?

- Daemon
  - Remote Managed Resources
  - Agent & MBean can survive the application & its JVM
  - Application must integrate remote communication services with the agent
    - RMI, SNMP, HTTP, …



JMX Deployment Models *[KregerHW 03]*

- Component
  - MBS is part of & controlled by the application
  - + access to fine grained instrumentation
  - - needs to be designed in strong relationship with the application
- Driver
  - Component with inversed authority role
  - MBS bootstraps the application & its MBeans
  - Ideal way of using JMX
  - Easy for application lifecycle management



JMX Instrumentation Pattern *[KregerHW 03]*

- Application is its own MBean
  - MyApplication implements…. MBean
  - Usually Standard, Open or Dynamic MBeans
  - MBean life-cycle usually bound to the application life-cycle & conversely
- Application is separated from its MBean(s)
  - 1 –n MBeans represent the application
  - Model MBeans are well suited
    - Especially due to the services they offer
  - All other (standard, dynamic, open) as well
  - MBean Life-cycle can be independent of the application (depends on who instanciates the MBean)
- *2 additional Instrumentation Patterns defined in [KregerHW 03]*
  - Publish-only
    - MBeans used to offer read-only data information to the management console with a well constrained overhead
    - Data caching is needed… ModelMBeans can do the work
  - Facades
    - MBean that aggregates interfaces to other (even remote) MBeans

JMX Dynamics (1)

- Managed Object Interface level
  - Dynamic MBean
    - Exposed management interface can be built at run time !
    - Full control of the interface by the MBean programmer
    - Invocation behavior can change over time
      - If (methodInvocation = "getStatus") then
        - if (lunchTime == true) return this.busy()
        - Else return this.getStatus()
  - Model MBean
    - Management interface is build incrementally
    - Behavior can be tailored (cache management , persistence, …)

JMX Dynamics (2)

- Remote dynamic loading of MBeans
  - Using the MLet service
  - Well designed for
    - delegation
    - Managament interface update

JMX Dynamics (3)

- Notification driven management orchestration
  - Every MO can be a notification emitter
    - Just become a notification broadcaster
  - Every MO can subscribe to notifications
    - Subsription on a per (source, notificationType) basis
  - Entire management behavior can be build on notification exchange
    - Even one MO can be a listener of its own broadcasted notifications and implement its behavior entirely in the listener

JMX Dynamics (4) : all together

- Fancy automated management scenarios are possible
  - Setup a listener on the create/delete & registration of MBeans
  - When a specific MBean appears
    - Remote load the corresponding monitoring MBean
  - Manage the application

This is Autonomous Management !

JMX Security (1) Access control

- Full part of JMX 1.2
  - Permission-based access control
- MBeanServer permission policies
  - What operations of the MBeanServer are allowed to which application
    - create an MBeanServer in/outside a MBeanServer factory
    - Release an MBeanServer
    - Find an MBeanServer
    - Get/set a builder
  - Grant given to applications
  - Implemented in the MBeanServerPermissionClass

```
grant MyApplicationArchive.jar
{
 permission javax.management.MBeanServerPermission "findMBeanServer,
 releaseMBeanServer";
};
```

JMX Security (1) Access control

- MBean level operations under permission policies
  - Instanciation, register/de-register
  - Instance & name search, get, querying
  - Add/remove NotificationListener
  - MO Attributes get/set
  - MO Metadata access (MBeanInfo)
  - Operation invocation
  - Class Loader reference
- MBeanPermission Target
  - ClassName of the MBean to which the policy applies
  - ClassMember to which the policy applies (attribute, operation)
  - ObjectName to which the policy applies

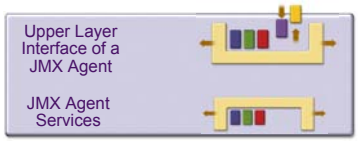
```
grant MyApplicationArchive.jar
{
 permission javax.management.MBeanPermission "StandardPrinter", "instanciate
  registerMBean";
 permission javax.management.MBeanPermission "StandardPrinter:reset", "invoke");
 permission javax.management.MBeanPermission "StandardPrinter:Status", "getAttribute");
 permission javax.management.MBeanPermission "StandardPrinter:Status[name="hp1"]",
  "setAttribute");
};
```


JMX Security (3)

- MBean Trust
  - Grant trust to MBeans signed by an authority
  - Only trusted MBeans can register in the server
- Security violation => SecurityException
- Uses the Java security services
  - Java Authentication & Authorization Service (JAAS)
    - User authentication
    - Access control to methods & objects
  - Simple Authentication & Security Layer API
  - Java Secure Socket Extension
    - TLS & SSL implementation

JMX Remoting : JSR 160

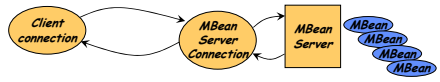
- Unclear JSR 03 (JMX) concerning the Client/Agent interactions :
  - (RMI) connectors under specified
  - operational model for remote listener
  - …
- JSR 160 (JMX Remote API) clarifies these points
  - public draft june 2003
  - applies to JMX 1.2



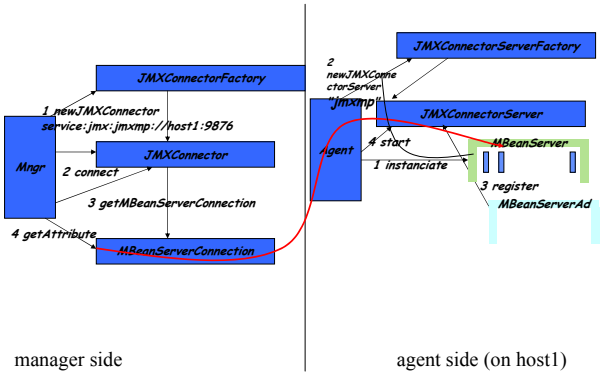
JSR 160's goals

- From the text:
  - "*interoperability*, *transparency*, *security*, and *flexibility*."
- *Standardizes* at least one protocol between client and agent (over RMI)

  ⇒ client independancy from agent implementation support (sun, MX4J, …)
- Remote client side operations mostly like local Mbean server interface ones

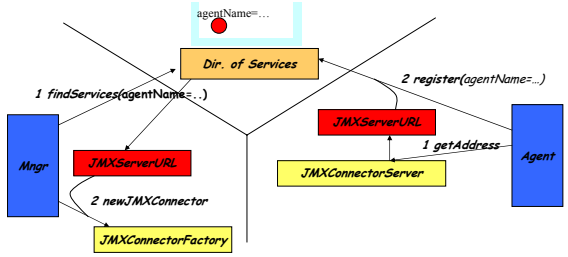


A matter of Connectors



Directories of (server) Connectors

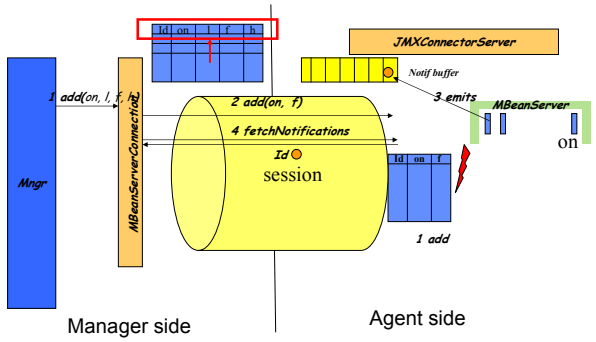
- No "JMX directory of services"
- Means to feed existing ones…
  - text mode (JMXServiceURL): SLP, JNDI/LDPA
  - Stub to JMXServerConnector: JINI, (JNDI/LDPA)
  - Basic attributes for (JMX) Services lookup.



Notification handling (1)

- Client/Server context: where are things done ?
  - Filtering: on agent (⇒serialization occurs)
  - Listeners on manager
  - Handback data on manager (usual case)
  - State data linked on Connector connection ("session")
- On client Add/remove Listener: <listener ref, filter ref, handbackdata ref, ObjectName>
  - Filter serialization ⇒ dedicated ListenerID management
- Implemention hints in JSR160: global Notification buffer linked to a given JMXConnectorServer
  - Remote fetch method in a JMXConnect (client) scope

Notification handling (2)



Implementations

- A mandatory RMI implementation
- A generic Connector: « pluggable transport »
- JMX Message Protocol (JMXMP)
  - Optional in conformant implementations
  - A « meta » transport protocol
    - Handshake phase and profil negociation (I.e: type of authentification)
  - Messages and interactions are clearly described
  - Object wrapping (serialization) is hookable
  - Not mandatory
  - Concrete sample implementation
    - Full duplex stream oriented communication interface over TCP
  - If you have time during your next vacation :
    - Do the SOAP implementation of JMXMP?

JSR 160 : Summary

- +++++
  - Very clear and precise specification
  - HTTP connectors are no longer part of the standard
  - Generic notification subscriber
    - Server connector subscribes to all possible MBean broadcasters that appear in a MBeanServer
- What is not detailled
  - Security aspects
    - Use of security subject for some interactions
  - Classes loading issues
    - On manager side (how to ensure which class loader does a JMXConnector use ?)
    - On agent side
      - MBean do not need to know all possible Filter class
      - How to ensure which class loader does a JMXServerConnector use ?
  - All we forgot ;-) …

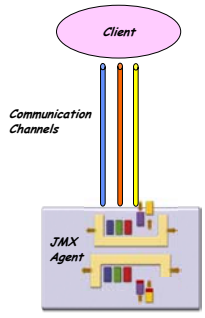
Usefulness for Dynamic Networks

- Connectors are MBeans + Mlet ⇒

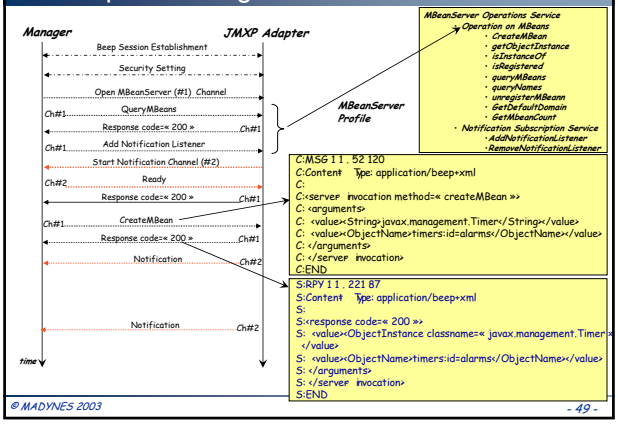
  Management infrastructure connectivity is easily configurable
  - E.g. dynamic download & setup of connectors can be context specific !
  - Still a bootstrap issue
    - RMI is there to solve the problem
- Better security support

JMX Remoting : JMXP *[draft-harold-jmxp00]*

- Application protocol to access JMX agents
- BEEP-based
  - BEEP profile
    - 3 profiles MBEANSERVER, MBEAN, NOTIFICATION
    - 3 allocated channels
      - MBEANServer interactions : CH1
      - NOTIFICATION interaction CH2
      - MBEAN interaction : CH3
  - authorization policies
    - Java 2 platform security
  - provisioning rules



A sample exchange scenario



The MBean Profile (Channel #3 operations)

- Get/Set values on MBeans Profile
  - Invoke Parameters :
    - Mbean instance name
    - (attributeName, [AttributeValue] *(set only)*)*
  - OK Response parameters :
    - Mbean instance name
    - (attributeName, Attribute Value)*
- MBeanInfo profile
  - Download a MBean metadata (MBean definition)
    - Parameter : target MBean instance name
- MBean Invocation profile
  - Invoke a method (operation) on a remote MBean
    - Parameters
      - MBeanInstance Name
      - Operation name (e.g. reboot)
      - Operation parameters
      - parameterName + Parameter Value

```
C:MSG 1 1 . 52 120
C:Content Type: application/beep+xml
C:
C:<mbean invocation mbean=« madynes:id=MyFirstMBean »
  operation = « setMaxCopies » >
C: <arguments>
C: <value><Integer>5</Integer></value>
C: </arguments>
C: </mbean invocation>
C:END
```

```
S:RPY 1 1 . 221 87
S:Content Type: application/beep+xml
S:
S:<response code=« 200 »>
S: <value><void></void></value>
S: </response>
S:END
```


Notification profile

- JMX Notifications are issued by a server to clients
- Clients register first
- Notification components
  - Notification description
    - metadata
  - Notification instance
    - values

```
S:ANS 2 23 . 652 98
S:Content-Type: application/beep+xml
S:
S:<notification type=« pc.alarm » >
S: <value>
S: <composite-data>
S: <structured-type name=«PCAlarmNotification » >
S: <item name="message" description="human readable text">
S: <scalar-type>String</scalar-type>
S: </item>
S: <item name="sequenceNumber">
S: <scalar-type>Long</scalar-type>
S: </item>
S: <item name="timeStamp">
S: <scalar-type>Long</scalar-type>
S: </item>
S: <item name="type">
S: <scalar-type>String</scalar-type>
S: </item>
S: <item name="notificationID">
S: <scalar-type>Integer</scalar-type>
S: </item>
S: </structured-type>
S: <member key="value">
S: <String>A PC related alarm</String>
S: <Long>42</Long>
S: <Long>102045873000000</Long>
S: <String>pc.alarm</String>
S: <Integer>13</Integer>
S: </member>
S: </composite-data>
S: </value>
S:</notification>
S:END
```

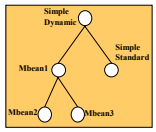

Encoding & security

- Scalar types
  - All OpenMBean like standard Types
    - Void, Boolean, Byte, Character, String, Long, …
- Structured types
  - OpenMBean types
    - Tabular
    - composite
  - Arrays
    - <array>
      - <value>….</value>
      - <value>…. </value>
  - Attribute (AVA)
    - <AttributeName, Value>
  - Composite
    - Simple or structured type (e.g. a struct), i.e. the description of the type
    - + member element (the value of the type)
  - Tabular
    - Simple or structured type
    - +one or more rows
  - Dedicated JMX types
    - MBeanInfo Value
    - ObjectInstance (class + ObjectName)
- Security Required profiles
  - MD5 (authentication)
  - TLS (confidentiality)
  - Combination and other algorithms for both authentication & confidentiality

JMX & Legacy

- Example of integration with legacy management
  - CMIS/JMX Gateway

JTMN CMIS Browser



JMXP : pros & cons

- An excellent initiative
  - A first proposal of a BEEP-based management protocol
- Notification subscription
  - Good to have it available !
  - Missing items
    - No Filtering, scoping in the JMXP parameters
    - « ***The Blues Brothers » syndrome***
      - No NotificationType based subscription in the current approach
        - Not even in JMX
      - Impossible to subscribe to the following
        - I am interested in all Alarms
      - Possible
        - I am interested in alarms from instance1, instance2, ….
  - How to deal with optional data in notifications ?
    - Change the metadata in each invocation.
  - Not part of the standard protocol so far ☺

JMX : + & -

- A marvelous wedding
  - Many Java*vores*, & some OSI*rists*
- Result:
  - OSI management without its limits
  - An Agent-toolkit
    - Trivial in usage (Ideal learning curve)
    - Easy to integrate (Especially if the application is in Java)
  - High quality OpenSource implementations
    - …especially MX4J et MC4J, TMX4J, …
  - Outstanding Documentation
    - Specs & books & examples
- Real Support of Dynamics
- Any limits ?
  - No neutral information model specification & language
    - Well : Java is one and a link to UML is always there
  - Agents (or at least the agent part of an application) are necessary in Java … I like that ☺
  - Client (Manager) side or at least remote access (remoting) took time to appear (JSR 160)
    - Now proprietary approaches are deployed (interoperability ?)
    - The specification took time but the standard is of great help and very well described

JMX future

Source : "JavaTM Management Extensions (JMXTM): Recent Changes and Ongoing Applications" TS 2274 presentation at JavaOne 2003 by Eammon Mc. Manus

- Excellent ideas to come
  - Metadata support
    - `@management` tag to automate the generation of the MBean interface
  - Hiearchical Management architecture
    - Proxy agent (proxy's remote MBeans in one JMXServer)
  - Reliable Notification Delivery

Tutorial Outline

- Growing Dynamics and their Impact on Management *(20 minutes)*
  - Dynamics
  - Collapsing time scales
  - The need for automation and autonomy
  - Management solutions for the dynamic world
  - Standardized approaches
  - Emerging alternatives
- JMX : Dynamic Management for/through the Java World *(60 min)*
  - Basic Concepts
  - Dynamic components
    - Attribute, method & notification level
    - Managed Object level
  - Remoting
  - Implementations
  - Application domains & instrumentation patterns
- SyncML-DM : an Approach for Managing Dynamic Devices *(60 min)*
  - Representation Protocol for Device Management
  - Device Management Protocol
  - Standardized Objects
  - Device Management Tree
  - Security for Device Management
- Case studies *(40 min)*
  - Ad hoc networks management
- Conclusion & discussion *(15 min)*

Synchronization in the early days (before 2000 A.D.)

Islands of disconnected Information

Device-Vendor-Product-Application specific synchronization:

Multiple devices (Phones, PDAs, WebTV)

Major Manufacturers (Alcatel, Nokia, Ericsson, …)

Several synchronization products (TrueSync, FoneSync, Intellisync Anywhere)

Server Applications (Lotus Notes, ODBC databases, yahoo…)

The Babel-Synchronization framework

Towards open data synchronization

2000: creation of the SyncML consortium (www.syncml.org).

Members : More than 600 companies-IBM, Nokia, Motorola, Ericsson, Matsushita, Symbian, Openwave …..

Objective : Develop an Open Standard for data Synchronization

Additional Results : Device Management Framework for managing devices .

Synchronization Markup Language is an Open Specification for universal synchronization

Use cases for Device Management

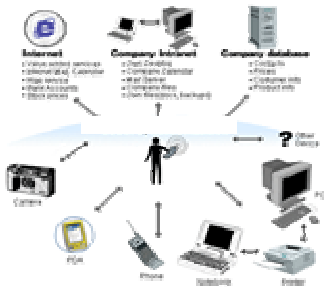
•Troubleshooting

- Remote service management
- Personal Management
- Storage Management
- Monitoring
- Software download

•Over the air Mass Configuration

Challenges in Device Management

- Heterogeneous devices
- Many applications
- Multiple network connectivity
- Limited resources



Requirements for Device Management

- Operate effectively over wireless and wired networks
- Support a variety of transport protocols
- Support arbitrary networked data
- Enable data access from a variety of applications
- Address the resource limitations of the mobile device
- Build upon existing Internet and Web technologies

Wireless/Wired Network Operations

Ubiquitous wireless access

1. High network latency
2. Limited bandwidth
3. Low reliability of both data and connectivity
4. Dynamic Addresses and network connectivity
5. Firewall, Nats – several management domains
6. Out of coverage factors

Redefine Management

1. Management might be to late
2. Manage efficiently (whenever it's required without wasting resources)
3. Connection Oriented management
4. Application level naming and addressing
5. Device initiated management
6. Different Fault behavior management

Transport Protocols

Several transport protocols are already used by wireless devices

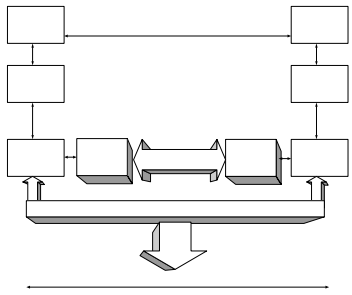
- HTTP (i.e. the Internet)
- WSP (the Wireless Session Protocol, part of the WAP protocol suite)
- OBEX (i.e. Bluetooth, IrDA, and other local connectivity)
- Pure TCP/IP networks
- Proprietary wireless communication protocols

All of them are connection oriented ……..

Multiple Applications

- Allow common data representation
  - Common personal data formats, such as vCard, todo etc.
  - Collaborative objects Relational data
  - HTML documents
  - Binary data
- Support extensions towards new data formats
- Programming Language Independent
  - No API (re) definition
  - No Communication Middleware dependency
  - XML Message exchanges as a glue

Scope of SyncML



XML based framework for data synchronization

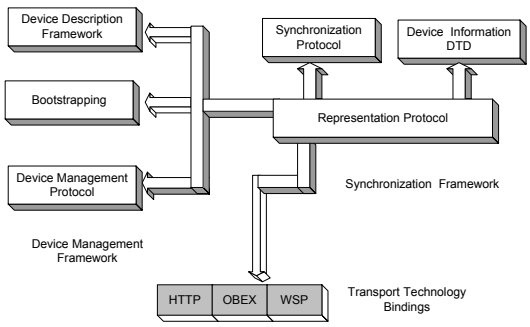
Message oriented data exchange protocol

Transport agnostic

Universal deployment

Extension for device management

SyncML Specifications – More than just XML

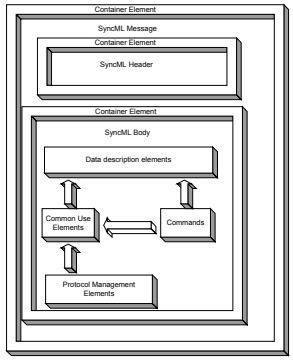


Representation Protocol - Objective

1. Specifies the structure of SyncML messages
2. Defines a vocabulary to describe management data and operations
3. Core component of the SyncML framework on which all others components are based

We will cover them in this order:

- Common Use Elements
- Protocol Management Elements
- Command Elements
- Data Description Elements
- Message Container Elements



Common Use Elements

Sub-elements of Command or SyncHdr used to provide a set of common functions

<Chal>……</Chal> Requesting a specific credentials
<Cmd>……<CmD> ASCII name of a command
<CmdID>……</CmdID> Identifier for a command
<CmdRef>……</CmdRef> ID of a command to which reference is made
<LocURI>……</LocURI> Identifies the target/source for an operation
<Target>……</Target> Where an operation applies
<Sources>……</Source> Data/Operation Source for an operation
<Final/> End of package flag

```
<Add>
<CmdID>1</CmdID>
<Item>
<Source> <LocURI>1</LocURI> </Source>
<Data>……</Data>
</Item>
</Add>
```

1. Chal
2. Cmd
3. CmdID
4. CmdRef
5. Cred
6. Final
7. Lang
8. LocName
9. LocURI
10. MoreData
11. MsgID
12. MsgRef
13. NoResp
14. NoResults
15. Source
16. SourceRef
17. Target
18. TargetRef
19. VerDTD
20. VerProto

Complete list of common use elements

Identification Mechanisms

Source : Who is starting the operation or from where data originates
Target : Where operations are performed or where data is copied/replaced/requested

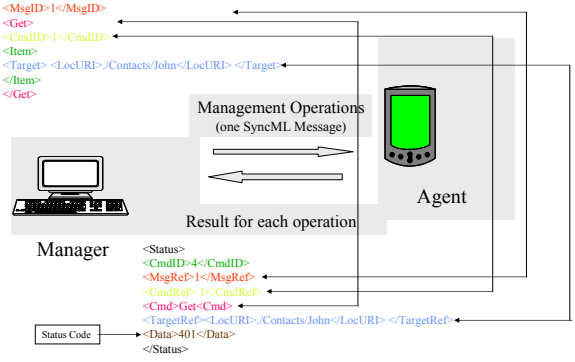
```
<Target>
    <LocURI>http://www.syncml.org/Server</LocURI>
</Target>
```

URI can represent :

1. a server
2. device
3. one data store <LocURI>IMEI:383/Rings</LocURI>
4. data item <LocURI>IMEI:383/Rings/ToneA</LocURI>

```
<Source>
    <LocURI>IMEI:383581748</LocURI>
</Source>
```


Protocol Management Element(s)



Status Codes (excerpts)

- Add
  - (200) OK : Command Completed successfully
  - (414) URI in command too long
  - (500) Command failed –generic code for failures
- Atomic
  - (507) Error occurred when performing an individual command
- Copy
  - (420) Device Full – no space left on device
  - (425) Permission Denied – no proper ACL
- Delete
  - (425) Permission Denied – no proper ACL
  - (403) Forbidden –node in use

1. Each command is associated to one or several status codes
2. If you know HTTP then you almost know the Status codes

Command Elements –functional classification

1. Data Command Elements – used to change application data

   Add - creates a new interior node
   Copy –copies values from a node to another at the client side
   Delete – deletes a node (and all its subnodes)
   Exec – process execution on the target
   Replace – overwites value for an existing node
   Get – retrieves data from the target

1. DataStore Command Elements – Actions for an entire datastore

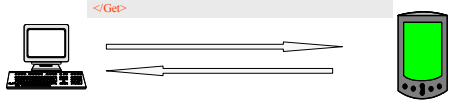
   Alert – used for notifications, text displays
   Results – contains results from a Get

2. Process Flow Commands – enhanced processing control

   Atomic – all subcommands must be executed
   Sequence – subcommands must be executed in order

The Duo: Get and Results

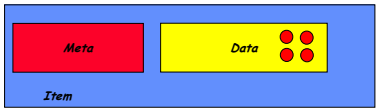
```
<MsgID>1</MsgID>
<Get>
<CmdID>2</CmdID>
<Item>
<Target> <LocURI>./Contacts/John</LocURI> </Target>
</Item>
</Get>
```



```
<Results>
<MsgRef>1</MsgRef>
<CmdRef>2</CmdRef>
<CmdID>2</CmdID>
<Item>
<Source> <LocURI> ./Contacts/John</LocURI> </Source>
<Data>Tel:01564433</Data>
</Item>
</Results>
```


Data Description Elements

1. <Data>….</Data> encloses SyncML payload data
2. <Item>….</Item>
   - isolates a command from the underlying data
   - Contains Data, Identification and Metadata
3. <Meta>…..</Meta> provides meta-information about the data
   - Type of the data
   - Size of the data



Meta Information

Meta-information related to a SyncML command or data item

```
<Add>
<CmdID>1</CmdID>

<Meta>
<Format xmlns='syncml:metinf'>chr</Format>

<Size>200</Size>
<Type xmlns='syncml:metinf'>text/x-calendar</Type>
</Meta>
<Item>
<Source><LocURI>./2</LocURI></Source>
<Data>

……

</Data>
</Item>
</Add>
```

Command to add a calendar entry

Tag indicating Meta Information
Data encoding is character (non-binary)

Size of the entry is 200 bytes
MIME calendar format for the entry

Location of the entry

Container Elements

- SyncML is message oriented
- Structure of a SyncML message is similar to HTTP
- Three types of containers:
  1. Message Container
  2. Header Container
  3. Body Container

```
<SyncML xmnls='SYNCML:SYNCML1.0>
<SyncHdr>
<VerDTD>1.0</VerDTD>
<SessionID>1</SessionID>
<MsgID>1</MsgID>
<Target>
<LocURI>http://www.syncml.org/Server</LocURI>
</Target>
<Source>
<LocURI>IMEI:383581748</LocURI>
</Source>
</SyncHdr>
<SyncBody>
<Get>
<CmdID>1</CmdID>
<Item>
<Target> <LocURI>./Contacts/John</LocURI> </Target>
</Item>
</Get>
</SyncBody>
</SyncML>
```

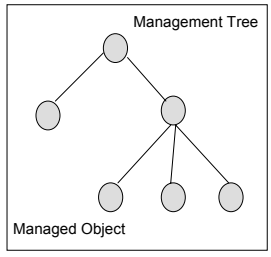
Message Container

Header Container

Body Container

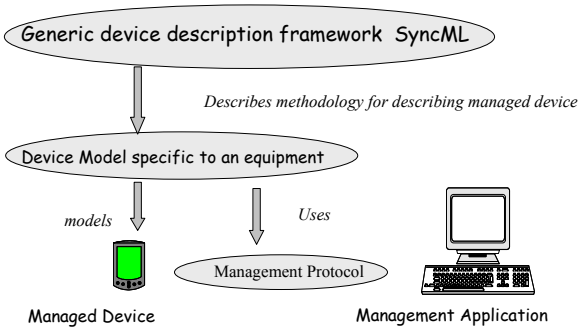
Device Management Framework

Describes the management information and how to access it



1. Framework for describing management information
2. Management Tree
3. Standardized Objects

Purpose of the Device Management Framework



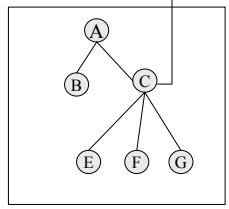
Modeling Management Information in SyncML

```
<!ELEMENT Node (NodeName, Path?, RTProperties?, DFProperties, (Node* | Value?))>
<!ELEMENT NodeName (#PCDATA)>
<!ELEMENT Path (#PCDATA)>
<!ELEMENT Value (#PCDATA)>
<!ELEMENT RTProperties (ACL, Format, Name, Size?, Title?, TStamp?, Type?, VerNo?)>
<!ELEMENT ACL (#PCDATA)>
<!ELEMENT Format (b64 | bool | chr | int | node | null | xml)>
```

Dynamic self-described management information

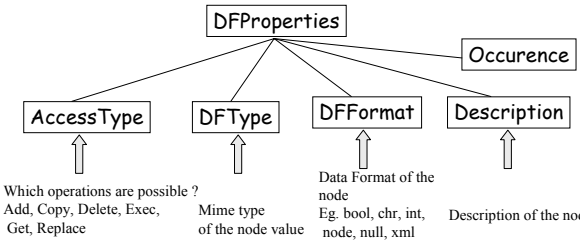
XML Tree Node captures information about :

- Name of the managed object
- ACL
- Subtrees
- Value
- Data Type of the Managed Object



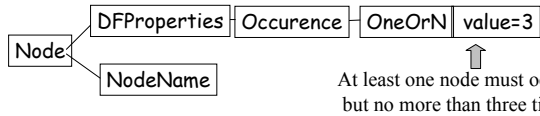
Management Information

Framework properties of a Managed Object



Framework properties do not change at run-time.
Analogous to a MIB2 definition

Information Modeling with framework properties

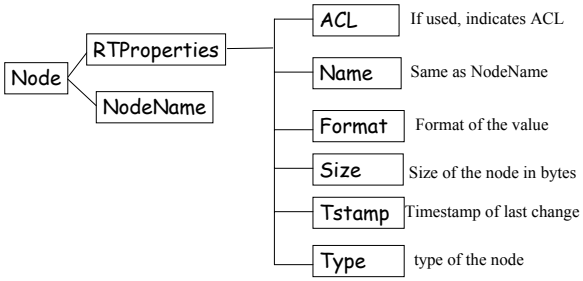


At least one node must occur, but no more than three times

Other possibilities:

- ZeroOrMore : Node can occur unlimited number of times or not at all
- ZeroORN : Node can occur up to N times, or not at all
- OneOrMore : Node can occur unlimited number of times, at least once

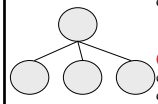
Runtime Properties of the Managed Object



Objective : Describe instantiated Managed Object

Runtime Properties vs Framework Properties

Instantiated MO



(Internal Node) Runtime Property *Type* Represents the name of a device description framework document.

(Terminal Node) Runtime Property Type equals *DFType* from the device description framework document

Runtime Property *Format* describes the data format of the node. Obviously must be equal to the *DFFormat* property

Vendor Document describing the Managed Object.

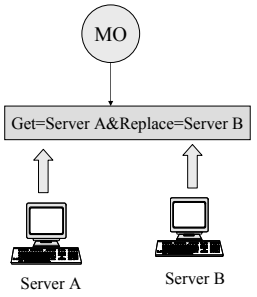
The Framework Property *DFType* indicates Mime type of the node

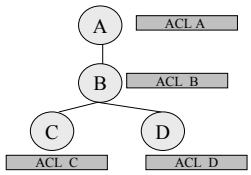
DFFormat indicates data format of the node

Do not confuse the two types of properties !

Access Control Lists (ACL)

ACL Property regulate access to a MO





ACL operation

1. Internal Nodes – access to ACL is regulated by the ACL itself
2. Terminal Nodes – Parent Node ACL regulates ACL modifications
3. If ACL empty then closest ancestor ACL is used

ACLs vs AccessType

- ACL is Runtime property – AccessType is a device framework property.
- AccessType specifies *what operations are supported* – the ACL specifies *who is allowed to perform these operations*

• ACL – AccessType interaction in case of dynamic (runtime created) nodes

		Direct Delete Command	Indirect Delete Command
Access Type	Delete allowed	Deleted	Deleted
	Delete not allowed	Not deleted	Deleted

Indirect delete occurs when an ancestor of the node has to deleted

Addressing Object Values and Properties

Addressing node values :
Object is identified by complete path to the root of the management tree
Example : *<LocURI>NodeA/NodeB</LocURI>*

Extended usage of the tags:
Meta, Format, Type

<Meta>
*<Format>*chr*</Format>*
*<Type>*text/plain*</Type>*
</Meta>

Meta to indicate metainformation
Format (string in this example) for the data format via *Runtime property Format*
Type =value for the Mime type

Addressing Property values : *node URI+?prop=<property_name>*
Example : addressing ACL property of Node B : */SyncML/NodeA/NodeB?prop=ACL*

Device Management Standardized Objects

Mandatory Device Management Objects for any SyncML device

Management Information Regarding :

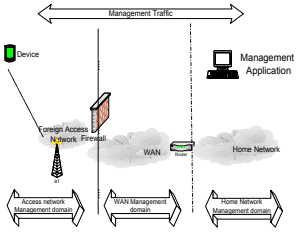
1. Connectivity information (protocol IPv4/IPV6, addresses, ports)
2. Accepted authentication methods
3. Bearer type (eg. Obex, GSM, CDMA)
4. Management server ID
5. Manufacturer ID,
6. Model and Device ID
7. Software version

Think MIB 2 for device management…..

Requirements for the Device Management Protocol

Requirement: Ability to deal with dynamic environments

- Dynamic network connectivity
- Unreliable communication medium
- Not always on-line devices due to out of coverage or limited mobility management
- Limited incoming connections for devices



Approach for the Device Management Protocol

Objective: Perform Device Management and allow user interaction in the management process

Features :

- Use the Representation Protocol
- Re-use security framework from the Synchronization Protocol
- Un-symmetric since only a server manages a client

Management Functionality :

- Read/Write/Replace Managed Object properties
- Create Managed Object
- Remove Managed Object

```
<Replace>
<CmdID>1</CmdID>
<Meta>…</Meta>
<Item>
<Target> 'URI of MO…..</Target>
<Data>'New value for MO'</Data>
</Item>
</Replace>
```

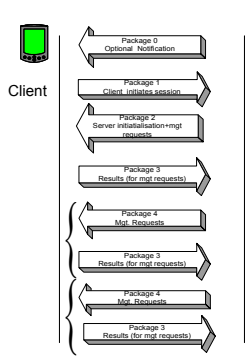
Design choices for the Device Management Protocol

1. Session Oriented
2. Client initiated
3. Out of band notification support
4. XML encoded
5. User interaction enabled

Rationale

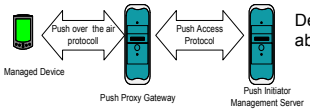
1. Devices might not be always online
2. Most firewalls will allow only device initiated connections
3. Notification support needed for "server initiated " sessions
4. User might have other priorities

A Management Session



- Five logical packages
- In most cases one package is a SyncML message
- Oversized packages can be included in several message
- Protocol state (each site) is determined by current protocol

Wireless Access Protocol based notification



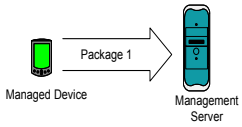
Access network technology specific

Device is provided information about :

- Required User interaction
- Background operation
- Server Identifier
- Session Identifier
- MD5 authentication
- Vendor specific

Package 0 Delivery

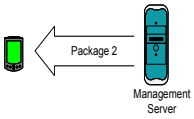
Client initialization



Information about :
- Protocol Version
- Session ID (equal to the notification)
- Maximum message size accepted by client
- Client credentials
- Client and Server IDs (URI)
- Initiator - is this a reply to a notification or a pure client initiated session ?

```
<SyncHdr>
<VerDTD>1.1</VerDTD>
<VerProtocol>DM/1.1</VerProtocol>
<SessionID>1</SessionID>
<MsgID>1</MsgID>
<Target>
<LocURI>http://www.server.com</LocURI>
</Target>
<Source>
<LocURI>IMEI:03839028</LocURI>
</Source>
<Meta><MaxMsgSize>4000</MaxMsgSize></Meta>
…
….
</SyncHdr>
<SyncBody>
<Alert>
<Cmd>1</Cmd>
<Data>200</Data>
</Alert>
```


Server initialization



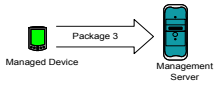
Information about :
- Protocol Version
- Session ID (equal to the package 1)
- Server credentials

Additional status information abou package 1 commands and header.

Additional commands

```
<SyncHdr>
….
<SessionID>1</SessionID>
<MsgID>2</MsgID>
<Source>
<LocURI>http://www.server.com</LocURI>
</Source>
<Target>
<LocURI>IMEI:03839028</LocURI>
</Target>
<Cred>
<Meta>
<Type>syncml:auth-basic</Type>
<Format>b64</Format>
</Meta>
<Data>  base64 formated userid:password</Data>
</Cred>
</SyncHdr>
<SyncBody>
<Status>
<MsgRef>1</MsgRef><CmdRef>0</CmdRef>
<Cmd>SyncHdr</Cmd>
<TargetRef>http://www.server.com</TargetRef>
<SourceRef>IMEI:03839028</SourceRef>
<Data>212</Data> (Remember error codes)
….
```


Client Response

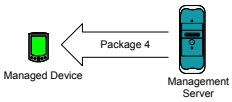


Information about :
- Session ID
- Server Authentication (package 2)
- Session ID (equal to the package 1,2)
- Status of management operations
- Results if package 2 contained a Get Request
- Last message in this package must contain the <Final> tag to mark the end of the package

```
<SyncHdr>
…….
</SyncHdr>
<SyncBody>
<Status>
<CmdID>5</CmdID>
<MsgRef>2</MsgRef>
<CmdRef>0</CmdRef>
<Cmd>SyncHdr</Cmd>
<Data>212</Data>
</Status>

<Results>
<CmdID>6</CmdID>
<CmdRef>1</CmdRef>
<MsgRef>2<MsgRef>
<Cmd>Get</Cmd>
<item>
<Source>..</Source>
<Data>Returned value</Data>
<Item>
</Results>
<Final/>
….
```

Management goes on




```
<SyncHdr>
……..
</SyncHdr>
<SyncBody>
……
<Replace>
<CmdID>8</CmdID>
<Meta>
<Format>b64</Format>
<Type>Mime type of MO</Type>
</Meta>
<Item>
<Target>
<LocURI>URI of MO</LocURI>
</Target>
<Data>B64 encoded value for
the MO
</Data>
</Item>
<Final/>
….
```

1. Status information about message header in package 3
2. Additional Commands
3. Format and Type of the managed object (MO)
4. Last message contains the <Final> tag

Bootstrapping

Basic initial configuration of the "Management" agent and transport connection

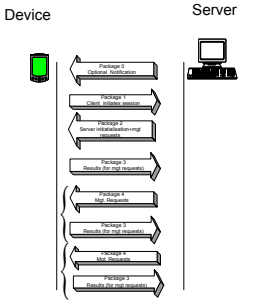
Existing Approaches :

- Out of band signaling (WAP Push)
- SIM card
- Pre-programmed

SyncML Bootstrap Protocol :

Provides Management Server address to Client
Authentication settings (Proof of Server ID)
Network level configuration setting for the communication (TCP/HTTP)

Security in SyncML Management

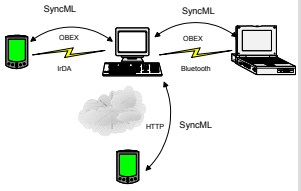


1. Authentication of Server
2. Integrity check of the message (MD5/basic authentication)

1. Two way authentication
2. Integrity check of the message
3. Access Control List mechanism
4. Confidentiality provided by transport level (HTTPS, SSL, OBEX)

Manage you wherever you are

- Ubiquitous management protocol
- Transport independent
- Bindings defined for short-range communication technologies as well as over IP/HTTP connections




```
POST ./servlet/syncit HTTP/1/.1
Host: www.mydevice.com
Content-Type :
application/vnd.syncml+xm;
Charset="utf-8"
Content-Length: 1023
Accept: application/vnd.syncml+xml

<SyncML>
<SyncHde>
…..
….

</SyncML>
```

SyncML /HTTP example
Simple Idea : Embed any SyncML message in a POST request

Existing Implementations

Proprietary implementations :

Major actors and contributors to the standard.

Open Source :

Sync4J (http://sync4j.sourceforge.net) for the data synchronization.
No device management functionality.

LORIA/MADYNES :

SyncML toolkit available :
http://www.madynes.org/software.html

SyncML DM summary

- SyncML DM is more than just XML based configuration
  - Framework for describing management information
  - A set of standard Managed Objects
  - Transaction oriented Management Protocol
  - Network level transport agnostic
  - Flexible Access Control
- Adaptation of conceptual design approaches from the SNMP framework towards device mobility
  - Similar building blocks to the SMI, MIB2 and SNMP and Get/Set semantics
  - Reversed roles : a managed Device initiates a management session
  - Power for the users : allow/deny management actions
- What to expect
  - More OpenSource implementations
  - Interoperability studies
  - Performance analysis and large scale deployment tests

Menu

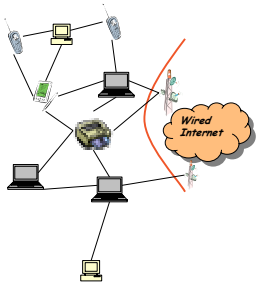
- Growing Dynamics and their Impact on Management *(20 minutes)*
  - Dynamics
  - Collapsing time scales
  - The need for automation and autonomy
  - Management solutions for the dynamic world
  - Standardized approaches
  - Emerging alternatives
- JMX : Dynamic Management for/through the Java World *(60 min)*
  - Basic Concepts
  - Dynamic components
    - Attribute, method & notification level
    - Managed Object level
  - Remoting
  - Implementations
  - Application domains & instrumentation patterns
- SyncML-DM : an Approach for Managing Dynamic Devices *(60 min)*
  - Representation Protocol for Device Management
  - Device Management Protocol
  - Standardized Objects
  - Device Management Tree
  - Security for Device Management
- Case studies *(40 min)*
  - Ad hoc networks management
- Conclusion & discussion *(15 min)*

Case Study of Dynamic Networks
Ad Hoc Networks

1. ANMP
   1. SNMP clustering
2. GMA
   1. Clustering & active code
3. Policy-based approaches
   1. COPS-PR Clustering for QoS provisionning
   2. A node-to-node policy exchange model

Ad hoc networks

- Autonomous wireless networks
  - No infrastructure (predefined)
  - No pre-configuration
  - With OR without connectivity to a fixed network
- Dual role entities
  - Any entity is both a terminal and a router
- Dynamic features
  - Limited & variable bandwidth
  - Power availability
- Why accept management ?
  - Entities in the network are volunteers to interact and provide the best possible service



Ad hoc networks dynamics & Management Requirements

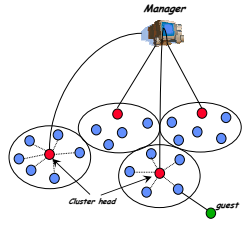
- Topology & connectivity
  - Connectivity among elements
  - Connectivity with an Internet Gateway
- Element capacity & constraints
  - Power variation & constaints over time
  - Limited CPU & storage capacity
- Management Requirements
  - Automation
  - Dynamic Configuration/Reconfiguration
  - Varying management roles & capacity in participating entities
    - From minimal participants (e.g. sensors, PDAs)
    - To full fledged management entities (e.g. gateway, PC-based terminal, …)
  - Management needs to be lightweight
    - Management overhead needs to be balanced against its cost
    - Limit the management trafic & processing
    - Efficient management
  - Security (including a secure management plane)
  - Management of a timely limited (temporary) world
  - Poor connectivity & high packet loss
    - Management plane must be robust
  - Some traditional faults are NOT necessarilly faults anymore
    - E.g. disappearing node

ANMP : Ad hoc Network Management Protocol *[ChenJS 00]*

- Focused functions
  - Configuration management
    - Network partitions, merge
    - Node connect, disconnect, Join, leave, death, birth, power on/off,
  - Fault Management
  - Monitoring (Data collection)
    - Manager rooted Spanning tree for polling
    - Trap-based update (on change data propagation)
      - Limit the communication on data change

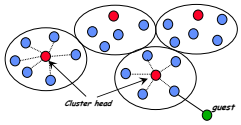
ANMP architecture & solution

- 3 level management hierarchy
  - Top-level Manager
    - Can be an agent towards a wired manager
  - Cluster heads (1st level agents & Managers)
- SNMP compatibility
  - Data is exchanged via SNMP compatible PDUs
  - Managament information is located in SNMP MIBs
- Clusters of neighbors
  - 2 clustering algorithms
    - Graph-based clustering algorithm
      - One hop neighbors (direct)
    - Geographical clustering algorithm
      - Up to 3 hop neighbors in a cluster



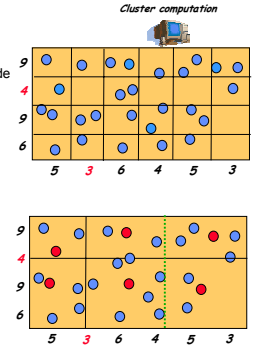
ANMP : Graph-based clustering

- Distributed clustering algorithm & protocol
  - Link between 2 nodes = in transmission range
  - Clusterhead = node with the smallest ID
    - ID is a unique identifier
      - Static :
        - MAC @,
        - Serial Number
      - Dynamic :
        - EnergyLevel + MAC@
        - IP@
- Node internal data structures
  - List of neighbors,
  - list of nodes in the cluster,
  - Ping counter (last time, logical, a ping was received from the cluster head)



Geographical clustering algorithm

- 2 level clustering algorithm
  - Node positions are known through GPS
  - A centralized / Periodical processing for cluster definition
  - A distributed processing algorithm to manage node mobility
- Central periodic clustering algorithm
  - Periodically initiated by the Manager
    - Calculation of stripes
    - Valley selection & division
    - Max distance for a box = 3x transmission range
    - Density adjustment facilities
  - Node density sensitive
    - min, max # of nodes / cluster
- Distributed mobility processing algorithm
  - Done at the cluster level
    - Leaving node send disjoin when leaving a cluster
    - Joining node sends Join message to cluster head when joining a cluster
    - Guest facility in each box for nearby unmanaged nodes
    - If node density changes (over nax, under min) recompute clustering

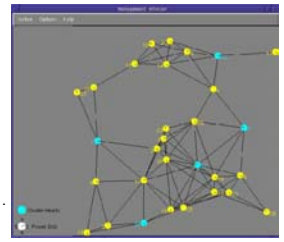


ANMP Management Information Base

- Power Usage MIB Group
- External Power : Y/N
- Remaining Power (watts-hour)
- Current Power mode
- Power mode costs
  - Power mode table (sleep, idle, active, …)
  - Consumption in each mode
- Battery
  - Age, type, serial#, last recharge time
  - Drain model (description, function)
- Radio Interface Power table
  - 1 Entry / interface
  - 32 bytes transmit/receive power
  - Power consumed in idle mode
  - Power consumption in sleep mode

ANMP Management Information Base & applications

- Topology group
  - Neighbor table
    - List of IP@ of all direct neighbors
  - List of supported topology protocols
    - Clustering protocols : graphical, geographical
  - # of current used topology maintenance protocol
  - Cluster protocols specific data
    - clusterID, #nodes in the cluster, …
- Applications
  - MIB 2 monitoring
  - Topology discovery
  - Power monitoring & management
    - E.g.K forcing a device to stay in alive mode



ANMP Agent Information

- Maintained by the Manager & each cluster head
  - Information Update Interval (ms)
  - %managed agents, %managed agents updated info available
  - Data collection tables (control + data tables)
    - Managed nodes power information
      - Control : number of samples stored, to be measured, sampling interval,
- M2M notification communication facility
  - Manager configures events to be sent by the cluster heads
  - Manager delegates monitoring (e.g. threshold monitoring) to clusterheads, through setting the control tables
  - Clusterhead selected alarm can lead to event issuance to the manager

ANMP Security

- Level Based Access Control Model (LACM)
- Each node has a Clearance Level
  - The Higher the level the lower the security access
    - 1 : access granted to manager with level 0 clearance certificates
    - 4 : access granted to manager with level 0,1,2,3
  - Clearance Ids are certificates
  - Clearance Ids are distributed by the manager to the nodes
- Data (mib objects) is grouped into projects
  - Each project (set of managed objects) has a clearance level which defines who can access it
- LACM effect on clustering
  - Cluster heads should have clearance to access managed nodes data
  - If Not, they are always bypassed by the node
    - Node encrypt data so it can only be decrypted by the manager

ANMP : Summary

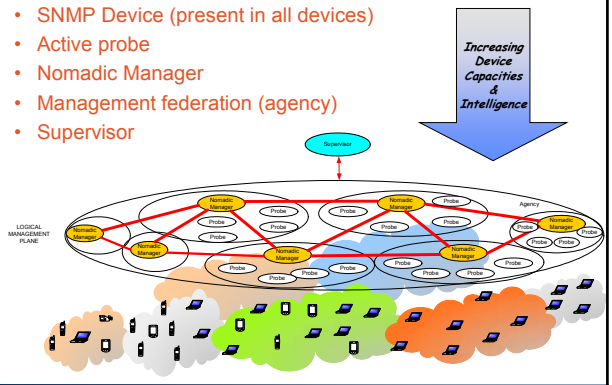
- A very complete SNMP compatible management architecture proposal
  - Good information models
- A first identification of a need for some active code facility
  - ftp download of code,
  - Association of the entry point of the code (function name) to alarms or event entries
- An outstanding contribution to the field
  - Interesting Clearance level based security model
- Very good coverage of mobile ad hoc nodes
  - >90% cluster-level managed nodes
- - :
  - No configuration transfert among clusterheads in cluster formation
    - Everything is conditioned to manager operations
  - Multicasting service usage not clear in the context of management
  - No clear definition of the right cluster size and its relationship to the management operations
    - E.g. monitoring frequency
  - Limited management functions
    - Topology discovery, + power status & management

Guerilla Management Architecture *[ShenSJ 02]*

- Management Architecture for ad hoc networks
- Goals : make management operational in highly dynamic networks
  - Maintain connectivity in the management plane within a managed ad hoc network
  - Minimize Management costs on :
    - CPU
    - Memory
    - Power consumption
  - Enable Disconnected Management !
- Approach
  - « Context-Aware » Self-configuration of the management plane
  - Hierarchical Management Framework
  - Management agency : a set of agents can speak as an agent for any other
  - Dynamic code deployment on active probes

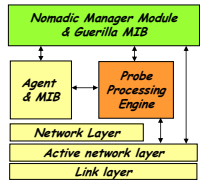
GMA : Architecture

- SNMP Device (present in all devices)
- Active probe
- Nomadic Manager
- Management federation (agency)
- Supervisor



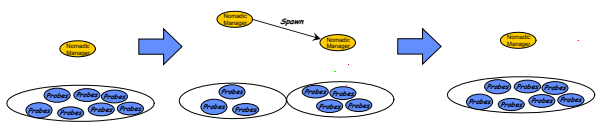
GMA : Component details

- Active Probe
  - Extension to the SNMP Agent
  - EE for hosting monitoring scripts
    - Both local monitoring and surrounding simple SNMP devices
  - Deployed by Nomadic Managers
- Nomadic Manager
  - Maintains connectivity in the Management Plane with other Managers
  - Decides to update the management plane configuration
    - E.g. can spawn new managers in other nodes
  - Holds management policies (rules) from the Supervisor
  - Maintains a Guerilla MIB
    - Aggregation of data collected from the active probes
- Supervisor
  - The manager



GMA : Protocol components

- Connectivity & management plane topology
  - Nodes send beacons (hello's)
  - Nomadic Manager decides of its coverage domain
    - Can spawn a new nomadic manager to split a domain
    - Can rely on the Adaptive Dynamic Backbone clustering protocol for self-configuration of domains
      - Builds trees of nodes rooted in an Nomadic Manager
  - Maximize a utility function to select the management function to perform
    - U = f(powerUsage,CPU Load, ManagementGain, …)



GMA : Management plane clustering algorithm

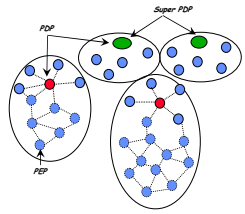
- Relies on the CLTC algorithm
  - Initially a power management protocol
    - k connectivity with minimized power consumption power assignment
  - Cluster-based Topology Control
    - 2 level topology control
      - Intra-cluster
      - Inter-cluster heads
    - 3 phase power level calculation
      - 1 : clusters establishment
      - 2: intra-cluster power assignment
      - 3 : inter cluster power assignment
        - K connectivity enforcement among clusters

GMA : Summary

- A MbD approach
  - Topology & power dynamics
  - Code mobility
- Heterogeneous Devices considered
  - ManagementRole = f(Device capacity)
- Fundamental Good Assumptions
  - Self-organization of the management plane is necessary
  - Disconnected management can be done
- Limits
  - So far no value-added management service deployed on the framework
  - Except the management of the management plane topology itself
  - Scalability issues are not convincing yet
- Open Questions
  - Why do we still need a manager ?

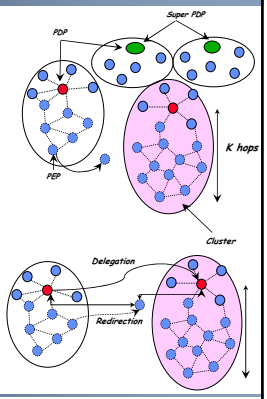
Policy-based Approaches *[Phanse 02]*

- Context : QoS management in the ad hoc plane :
  - Resource usage authorization
  - Policy-based admission control
  - Dynamic bandwidth allocation
  - Differentiation provisioning
  - Monitoring
- Approach
  - Decentralized COPS-PR
    - Hybrid Provisionning and Outsourcing model
  - Clustering & Cluster Management
  - Dynamic Service Redundancy
  - Service Location Discovery (find the PDP)



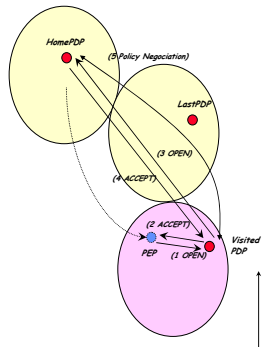
Clustering

- K-Clustering
  - Maximum K hops away from a policy server
  - A client can be served by a server that is less than k-hops away
- Initial Deployment
  - Super PDPs are present in the network
- Redirection service
  - A server (PDP) can dynamically redirect a client to a less distant PDP
  - Redirection is Server initiated
- Delegation Service
  - A PDP can delegate part of its decisions to another PDP



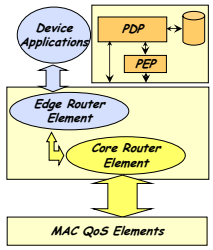
Extensions

- Multi-domain policy negociation
  - Inter-PDP signaling
  - Home PDP + Last PDP fields in COPS messages
  - If no policies found by the visited PDP => negociation with the HomePDP
  - Negociation = Policy download
- Possible improvements (OF)
  - Policy transfert during redirection
  - PEPs could « come with » their own policies



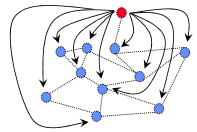
Node-to-node policy exchange approach *[MunarettoAF 02]*

- Context : Corporate ad hoc networks
  - Uniform management & company wide policies
- Management function
  - Diffserv-like QoS management in the ad-hoc network
- Approach
  - Every *ad-hoc* device exports both functions
    - Edge router (marking, metering, policing, shaping)
      - Acts on local applications
    - Core router (classifying, schedulling,…)
      - Acts on routed packets
- Management Plane
  - Each *ad-hoc* device embeds both
    - PDP + Policy Repository
    - PEP

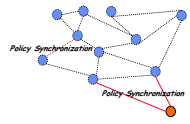


N2N addressed problems *[MunarettoAF 02]*

- 2 phases policy distribution & synchronization
  - 1: feed the PIBs of connected devices
    - Manager is a PDP
    - Ad-*hoc* nodes are in a PEP role
  - 2: permanent synchronization among 2 meeting devices
    - Applied to new entrants
- Policy authority
  - Management Hierarchy with delegation
  - Policies are tagged by managerID
  - Each managerID has an authority level known by all
  - The more authoritative policy always replaces less ones



Phase 1: Policy distribution by Prime network management



Phase 2: Policy snchronization

N2N Policy exchange approach

- Summary
  - A focused approach
    - Corporate
  - An interesting approach
    - Device level Policy Management Framework
- Current limits
  - Synchronization protocol not described
  - Some open problems on policies consistency
  - No description of addressed policies & policy levels
  - Hierarchical authority model is not sufficient to cover security issues at all !

Ad hoc management summary

- Other approaches
  - MARE
    - Used mainly for service location
    - Mobile agents + distributed tuple-space
- Common vision of the problems
  - Management overhead must be minimized
- Similar approaches
  - Clustering (various algorithms)
  - Delegation (agents, active code, extensible SNMP agent, …)

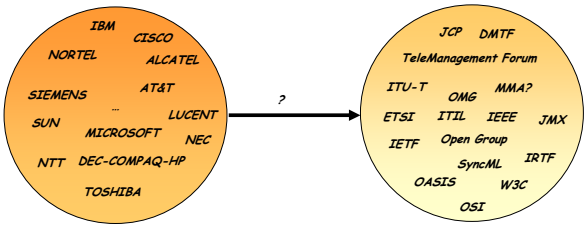
Menu

- Growing Dynamics and their Impact on Management *(20 minutes)*
  - Dynamics
  - Collapsing time scales
  - The need for automation and autonomy
  - Management solutions for the dynamic world
  - Standardized approaches
  - Emerging alternatives
- JMX : Dynamic Management for/through the Java World *(60 min)*
  - Basic Concepts
  - Dynamic components
    - Attribute, method & notification level
    - Managed Object level
  - Remoting
  - Implementations
  - Application domains & instrumentation patterns
- SyncML-DM : an Approach for Managing Dynamic Devices *(60 min)*
  - Representation Protocol for Device Management
  - Device Management Protocol
  - Standardized Objects
  - Device Management Tree
  - Security for Device Management
- Case studies *(40 min)*
  - Ad hoc networks management
- Conclusion & discussion *(15 min)*

Conclusion (1)

- Dynamics is a growing parameter in the management plane
  - Dynamics is everywhere
  - The research community is starting to address the problem
- Ad-hoc management is one example
  - Few initiatives so far
  - Similar approaches
    - Clustering
    - Delegation
- Some approaches enable dynamic management
  - Or at least some parts of it
- Other initiatives contribute as well to the field
  - E.g. Debussmann's auto-instrumentation proposals
  - Jini-based federations of autonomous management architectures
  - …

Conclusion (2): Is the answer in standards ?



- How many standardization Consortia can be built from n proprietary solutions, given that a standard requires at least 2 companies to be established.
- Be careful, everything you standardize today becomes the legacy of tomorrow !

Conclusion (3) : The management future according to MADYNES

- Management will not disappear !
  - At least we hope it will not.
- The management future will be a mixture of :
  - (option 1) management of static only parts
  - (option 3) smart & automated management techniques
- Efforts must be maintained in both directions
  - Standards can play a role in both options as well
  - But are not the overall success condition
- Research may be more "sexy" in option 3
  - Covers & integrates potentially many fields & techniques

References

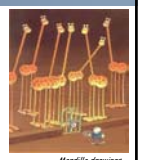
[Jakobson 01]	G. Jakobson, Management of Dynamic Networks and Services : Correlation-based Solutions, Invited Presentation, IFIP DSOM'2001, october 2001, Nancy, France.
[ChenJS 00]	W. Chen, N. Jain, S. Singh, ANMP : Ad hoc network Management Protocol, IEEE JSAC, August 1999.
[ShenSJ 02]	C.C. Shen, C. Srisathapornphat, C. Jaikaeo, An Adaptive Management Architecture for Ad Hoc Networks, IEEE Communications Magazine, February 2002, pp 108-115.
[Phanse 02]	K. Phanse, Policy-Based Quality of Service Management in Wireless Ad Hoc Networks, Preliminary Research Document for the Ph.D. Degree, Faculty of the Virginia Polytechnic Institute and State University, 2002
[draft-harold-jmxp]	W. Harold, Java ™ Management Extensions Protocol, draft-harold-jmxp-00, may 2003.
[MunarettoAF 02]	A. Munaretto, N. Agoulmine, M. Fonseca, Policy-based Management of Ad Hoc Enterprise Networks, HPOVUA 2002?
[KregerHW 03]	H. Kreger, W. Harold, L. William, Java and JMX : Building Manageable Systems, Addison Wesley, 2003.
[Hansmann02]	U. Hansmann, R. Mettala, A. Purakayastha, P. Thompson, SYNCML : Synchronizing and Managing Your Mobile Data. Prentice Hall, 2002
[RepProt02]	SyncML Representation Protocol v1.1. SyncML Forum (www.syncml.org), 2002
[RepDevProt02]	SyncML Representation Protocol, Device Management Usage SyncML, Forum (www.syncml.org), 2002
[DevManagProt02]	SyncML Device Management Protocol,SyncML Forum (www.syncml.org), 2002
[SyncMLNotif]	SyncML Notification Initiated Session, SyncML Forum (www.syncml.org), 2002
[SyncMLBootstrap]	SyncML Device Management Bootstrap, SyncML Forum (www.syncml.org), 2002
[SyncMLDevTree]	SyncML Device Management Tree and Description Session, SyncML Forum (www.syncml.org), 2002
[SyncMLObjects]	SyncML Device Management Standardised Objects, SyncML Forum (www.syncml.org), 2002
[SyncMLBinding]	SyncML HTTP Binding, SyncML Forum (www.syncml.org), 2002

That's all folks *[WarnerBros☺]*

Conclusion (4) : advice

To young researchers & Ph.D. students in the discipline

- Management needs **NEW** models and techniques
- Be creative
  - Do not stick to standards and implementation details
  - Free your mind !
  - Incremental research as a backup
- Validate your ideas
  - Models exist, environments too
  - Formalize your protocols
  - Simulate their behavior
  - Evaluate the performance
  - Implement when sounded
- Cooperate
  - There is a management discipline with an associated community of some very pleasant, motivated and competent people
- Be hungry of history
  - IM, LISA, DSOM, NOMS, MMNS,
  - *« The one that does not know where he comes from , does not know where he is heading to because he does not know where he is ! » from ????*



Mordillo drawings